# Introduction to Computing – Lab

Faculty of Information Technology & Computer Science

| Lab 07 |
| --- |
| **Course Instructor**: Afham Nazir |
| **Topics**: Loops (For Loop, While Loop, Do While Loop) |

## Instructions:

- Create separate C++ source files for each task, named as “task1.cpp”, “task2.cpp”, and so on depending on the task number.
- After completing all tasks, place all .cpp files into a folder. Name the folder with your university registration number (e.g., L1F20BSCS0999). Compress the folder into a .zip file and upload it on the portal.
- Ensure that the work you submit is entirely your own. Avoid copying from peers, online sources, or any other unauthorized material. Plagiarism will not be tolerated.
- If you encounter difficulties, feel free to reach out to the instructor. Collaboration and discussion are encouraged, but the final implementation should be your own work.
- Write clean and well-structured code. Use comments to explain key sections of your code to make it easier for others (and yourself) to understand.
- These tasks are designed to help you strengthen your logical thinking and problem-solving skills. Think through each problem carefully before starting to code. The aim is to develop a deep understanding of the problem and to devise solutions independently.
- Learning programming is about practice and perseverance—genuine effort in solving the problems will contribute significantly to your learning.

## For Loop

The for loop is typically used when the number of iterations is known in advance. It contains three main parts within its syntax: initialization, condition, and update, all defined in one line.

```
for (initialization; condition; update) {
    // Code to execute
}
```

**Example:**

cpp

```cpp
for (int i = 0; i < 5; i++) {
    cout << i << " ";
}
// Output: 0 1 2 3 4
```

## While Loop

The while loop is a pre-test loop, meaning it checks the condition before executing the loop body. It's typically used when the number of iterations is unknown, and it continues as long as the condition is true.

cpp

```cpp
while (condition) {
    // Code to execute
}
```

**Example:**

cpp

```cpp
int i = 0;
while (i < 5) {
    cout << i << " ";
    i++;
}
// Output: 0 1 2 3 4
```

## Do While Loop

The do...while loop is a post-test loop, meaning it executes the loop body at least once before checking the condition. It's useful for situations where the loop should run at least once, regardless of the condition.

```cpp
do {
    // Code to execute
} while (condition);
```

**Example:**

```cpp
int i = 0;
do {
    cout << i << " ";
    i++;
} while (i < 5);
// Output: 0 1 2 3 4
```

## Differences Between for, while, and do...while Loops

| *Feature* | for Loop | while Loop | do...while Loop |
|---|---|---|---|
| ***Condition Check*** | At the beginning | At the beginning | At the end |
| ***Execution*** | Executes if the condition is true | Executes if the condition is true | Executes at least once, then checks condition |
| ***Use Case*** | Known number of iterations | Unknown number of iterations | Ensure loop executes at least once |

## When to Use Each Loop:

- **for Loop**: Use when you know the exact number of iterations beforehand, such as iterating over arrays or counting within a range.
- **while Loop**: Ideal when the number of iterations is uncertain, but you know the condition that should stop the loop, like reading data until end-of-file.
- **do...while Loop**: Useful when you want the loop to execute at least once before any condition is checked, like in a menu-driven program where the options display at least once before checking for the exit condition.

---

## Task 1

Write a program that takes 10 integer inputs from the user, one at a time, and finds the maximum and minimum values using a for loop.

- **Example Input**: Enter numbers: 34 12 78 45 23 32 5 9 12 77
- **Example Output**: Maximum value: 78, Minimum value: 5

## Task 2

Take a positive integer as input and use a while loop to find and print the smallest and largest digits in the number.

- **Example Input**: Enter a positive integer: 52743
- **Example Output**: Smallest digit: 2, Largest digit: 7

## Task 3

Write a program that takes an integer as input and uses a while loop to calculate the sum of its digits.

- **Example Input**: Enter an integer: 3456
- **Example Output**: Sum of digits: 18

## Task 4

Write a program to print a series which starts from 1 and increase the 1 in next term till nth term, which is input by the user.

- **Example Input**: Enter input terms: 6
- **Example Output**: 1, 11, 111, 1111, 11111, 111111

## Task 5

Write a program to calculate the Least Common Multiple (LCM) of two positive integers using a for loop.

- **Example Input**: Enter two positive integers: 12, 15
- **Example Output**: LCM of 12 and 15 is: 60

## Task 6

Write a program that checks if a given number is an Armstrong number. For a 3-digit number, if the sum of the cubes of its digits equals the number itself, it's an Armstrong number. Use a while loop for the calculation.

- **Example Input**: Enter a 3-digit number: 153
- **Example Output**: 153 is an Armstrong number

## Task 7

Write a program to calculate the GCD of two positive integers using a while loop and the Euclidean algorithm.

- **Example Input**: Enter two positive integers: 48, 18
- **Example Output**: GCD of 48 and 18 is: 6

## Task 8

Write a program that takes an integer as input and checks if the number is a palindrome (reads the same backward as forward). Use a while loop to reverse the digits and compare it with the original number.

- **Example Input**: Enter an integer: 1221
- **Example Output**: 1221 is a palindrome

## Task 9

Write a program that takes an integer n as input and prints the first n terms of the Fibonacci series. The Fibonacci sequence is 0, 1, 1, 2, 3, 5, 8, etc., where each term is the sum of the previous two terms. Use a for loop for this calculation.

- **Example Input**: Enter the number of terms: 7
- **Example Output**: Fibonacci series: 0 1 1 2 3 5 8

## Task 10

Write a program that takes an integer n and prints all prime numbers up to n. Use a for loop to check each number from 2 to n and determine if it is prime (divisible only by 1 and itself).

- **Example Input**: Enter a number: 20
- **Example Output**: Prime numbers up to 20: 2 3 5 7 11 13 17 19

---